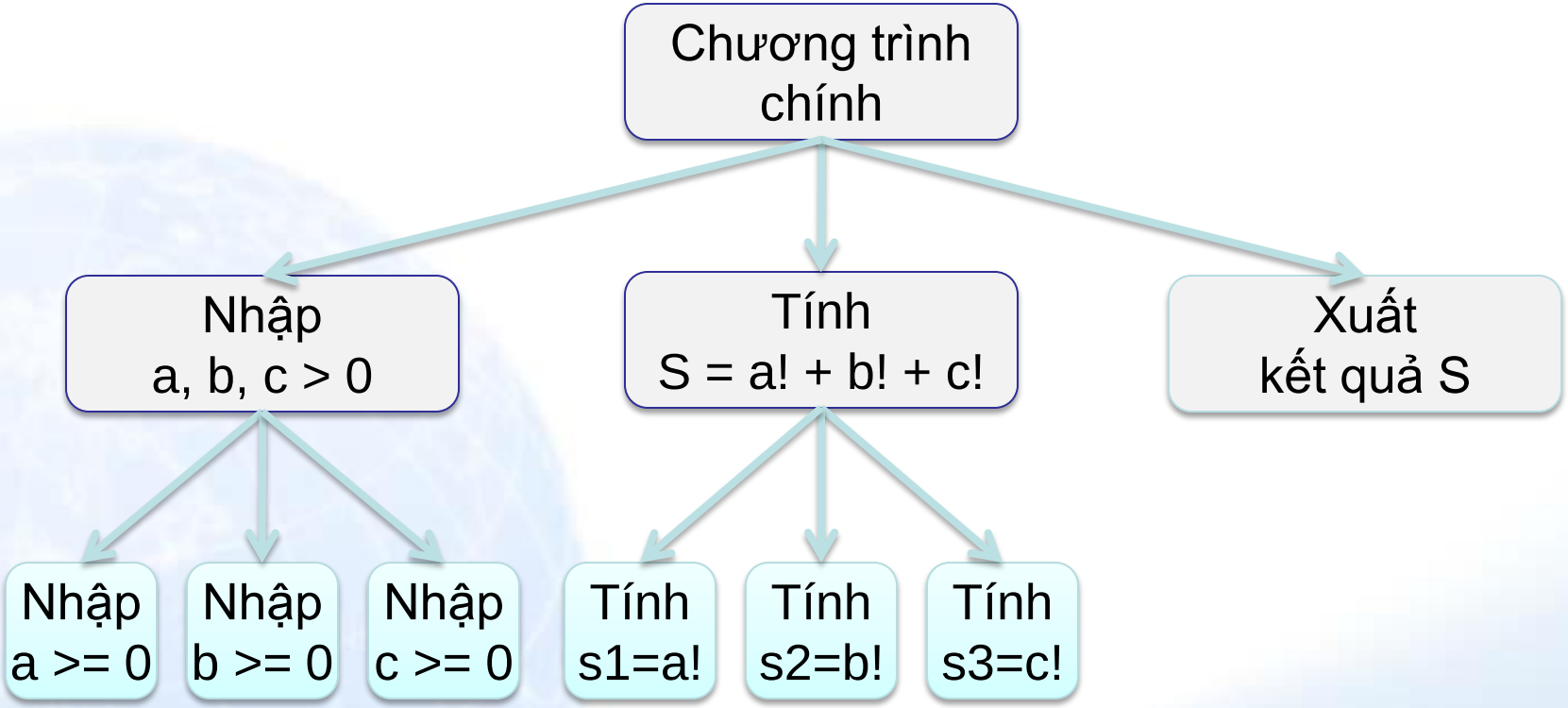
Chương 4
Hàm và cấu trúc chương trình

- Cấu trúc một chương trình
- Xây dựng và sử dụng hàm
- Truyền tham số con trỏ và địa chỉ
- Thuật toán tráo đổi giá trị 2 biến

Đặt vấn đề

- Viết chương trình tính S = a! + b! + c! với a, b, c là 3 số nguyên dương nhập từ bàn phím.



Đặt vấn đề

- 3 đoạn lệnh nhập a, b, c > 0

```
do {
      printf("Nhap mot so nguyen duong: ");
      scanf("%d", &a);
} while (a < 0);

do {
      printf("Nhap mot so nguyen duong: ");
      scanf("%d", &b);
} while (b < 0);

do {
      printf("Nhap mot so nguyen duong: ");
      scanf("%d", &c);
} while (c < 0);
```

Đặt vấn đề

- 3 đoạn lệnh tính s1 = a!, s2 = b!, s3 = c!

```
{ Tính s1 = a! = 1 * 2 * … * a }
s1 = 1;
for (i = 2; i <= a ; i++)
      s1 = s1 * i;

{ Tính s2 = b! = 1 * 2 * … * b }
s2 = 1;
for (i = 2; i <= b ; i++)
      s2 = s2 * i;

{ Tính s3 = c! = 1 * 2 * … * c }
s3 = 1;
for (i = 2; i <= c ; i++)
      s3 = s3 * i;
```

Đặt vấn đề

❖Giải pháp => Viết 1 lần và sử dụng nhiều lần

- *Đoạn lệnh nhập tổng quát, với n = a, b, c*

```
do {
        printf("Nhap mot so nguyen duong: ");
        scanf("%d", &n);
} while (n < 0);
```

- *Đoạn lệnh tính giai thừa tổng quát, n = a, b, c*

```
{ Tính s = n! = 1 * 2 * … * n }
s = 1;
for (i = 2; i <= n ; i++)
        s = s * i;
```

Cấu trúc chương trình

Khai báo thư viện hàm
Khai báo hằng số…
Khai báo hàm

Cài đặt tất cả những hàm con đã được khai báo

Gọi thực hiện các hàm theo yêu cầu của bài toán

1. **Chỉ thị tiền biên dịch:** giúp trình biên dịch thực hiện một số công việc trước khi thực hiện một số công việc trước khi thực hiện biên dịch chính thức

   VD: #include <stdio.h>
   #include <conio.h>

2. **Khai báo kiểu dữ liệu mới:** dung từ khoá **typedef**

   VD: typedef int songuyen;
   typedef float sothuc;

3. **Khai báo hằng và biến ngoài (nếu có):** khai báo các hằng số và biến ngoài dùng trong chương trình
4. **Khai báo hàm:** khai báo các hàm tự viết
5. **Chương trình chính**: hàm main là hàm bắt buộc trong chương trình. Hàm main có thể trả về giá trị kiểu nguyên (int) hoặc không trả về giá trị nào (void)
6. **Cài đặt các hàm:** viết chi tiết các hàm

Khái niệm hàm

- *Một đoạn chương trình có tên, đầu vào và đầu ra.*
- *Có chức năng giải quyết một số vấn đề chuyên biệt cho chương trình chính.*
- *Được gọi nhiều lần với các tham số khác nhau.*
- *Được sử dụng khi có nhu cầu:*
  - Tái sử dụng.
  - Sửa lỗi và cải tiến.

❖ **Chương trình con = Hàm (trong C)**

Đặc điểm của hàm

❖ Đặc điểu của hàm

- Là một đơn vị độc lập của chương trình.
- Không cho phép xây dựng một hàm bên trong một hàm khác.

❖Có 2 loại hàm

- **Hàm chuẩn:** Được định nghĩa sẵn bởi ngôn ngữ lập trình và được chứa vào các thư viện
- **Hàm tự định nghĩa:** Do người lập trình tự tạo ra nhằm đáp ứng nhu cầu xử lý của mình

Khuôn mẫu hàm

- Cần xác định các thông tin sau đây:
  - *Tên hàm.*
  - *Hàm sẽ thực hiện công việc gì.*
  - *Các đầu vào (nếu có).*
  - *Đầu ra (nếu có).*

Hàm

- Ví dụ 1: Xuất tổng của 2 số nguyên
  - *Tên hàm: XuatTong*
  - *Công việc: tính và xuất tổng 2 số nguyên*
  - *Đầu vào: hai số nguyên x và y*
  - *Đầu ra: không có*

```
void XuatTong(int x, int y)
{
        int s;
        s = x + y;
        printf("%d cong %d bang %d", x, y, s);
}
```

Hàm

- Ví dụ 2: Tính tổng của 2 số nguyên
  - *Tên hàm: TinhTong*
  - *Công việc: tính và trả về tổng 2 số nguyên*
  - *Đầu vào: hai số nguyên x và y*
  - *Đầu ra: một số nguyên có giá trị x + y*

```
int TinhTong(int x, int y)
{
        int s;
        s = x + y;
        return s;
}
```

Hàm

- Ví dụ 3: Nhập xuất tổng
  - *Tên hàm: NhapXuatTong*
  - *Công việc: nhập và xuất tổng 2 số nguyên*
  - *Đầu vào: không có*
  - *Đầu ra: không có*

```
void NhapXuatTong()
{
        int x, y;
        printf("Nhap 2 so nguyen: ");
        scanf("%d%d", &x, &y);
        printf("%d cong %d bang %d", x, y, x + y);
}
```

So sánh 2 hàm:

```
void XuatTong(int x, int y)
{
        int s;
        s = x + y;
        printf("%d cong %d bang %d", x, y, s);
}
```

```
int TinhTong(int x, int y)
{
        int s;
        s = x + y;
        return s;
}
```

Hàm nguyên mẫu (prototype)

<Kiểu dữ liệu> TênHàm([ds các tham số]);

Trong đó:

Kiểu dữ liệu trả về của hàm (kết quả của hàm/ đầu ra), gồm 2 loại

- ***void****: Không trả về giá trị*
- ***float / int / long / char */ kiểu cấu trúc / …*** *: Trả về giá trị kết quả có kiểu dữ liệu tương ứng với bài toán (chỉ trả về được 1 giá trị theo kiểu dữ liệu)*

Ví dụ:

```
int TinhTong(int a, int b);
```

Hàm nguyên mẫu (prototype)

- **TênHàm:** Đặt tên theo qui ước đặt tên sao cho phản ánh đúng chức năng thực hiện của hàm
- **Danh sách các tham số (nếu có):** đầu vào của hàm (*trong một số trường hợp có thể là đầu vào và đầu ra của hàm nếu kết quả đầu ra có nhiều giá trị - Tham số này gọi là tham chiếu*)

Cấu trúc chương trình khi có hàm

➢ **Cách 1:**

```
// khai báo thư viện
hàm1( ); //khai báo nguyên mẫu prototype
hàm2( );
// khai báo biến toàn cục nếu có
void main( )
{ // khai báo biến cục bộ
        …
        hàm1( ); // gọi hàm1
        hàm2( ); // gọi hàm2
        ...
}
hàm1( )   // khai báo chi tiết các hàm
{       // khai báo biến cục bộ
        <lệnh>;
}
hàm2( )
{       // khai báo biến cục bộ
        <lệnh>;
}
```

*Khai báo nguyên mẫu trước theo **cách 1** thì các hàm có thể gọi lẫn nhau.*

Cấu trúc chương trình khi có hàm

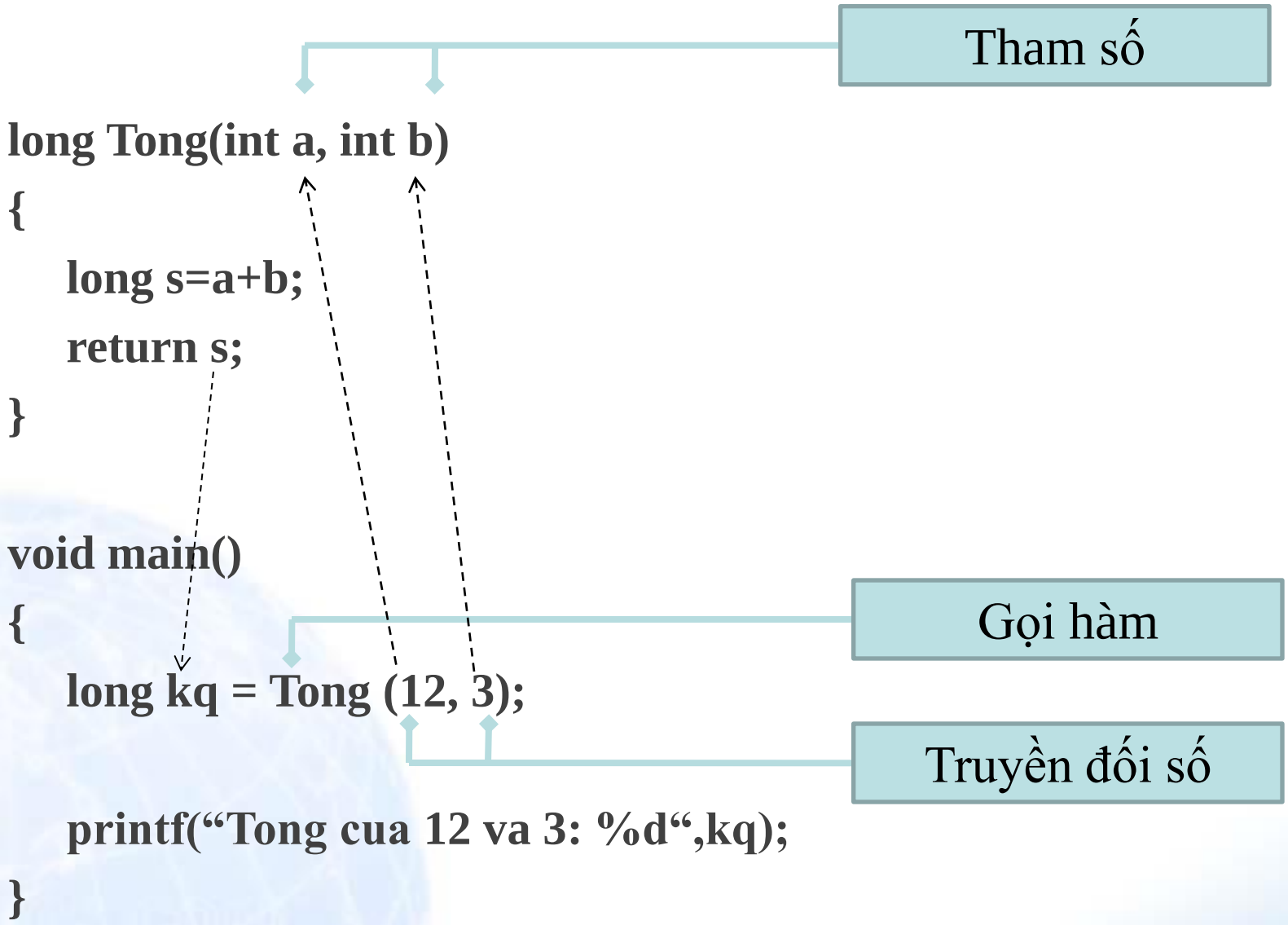
- **Cách 2:**

```
// khai báo thư viện
hàm1( )
{       // khai báo biến cục bộ
        <lệnh>;
}

hàm2( )
{       // khai báo biến cục bộ
        <lệnh>;
}

// khai báo biến toàn cục
void main( )
{       // khai báo biến cục bộ
        …
        hàm1( ); // gọi hàm1
        hàm2( ); // gọi hàm2
        …
}
```

Thường được sử dụng


Tham số
long Tong(int a, int b)
{
long s=a+b;
return s;
}
void main()
{
long kq = Tong (12, 3);
printf("Tong cua 12 va 3: %d",kq);
}
Gọi hàm
Truyền đối số

Hàm không trả về giá trị

❖ **Cài đặt**

```
void TênHàm([danh sách các tham số])
{
    Khai báo các biến cục bộ
    Các câu lệnh / khối lệnh hay lời gọi đến hàm khác
}
```

❖ **Gọi hàm**

TênHàm(danh sách tên các đối số);

Những phương thức loại này thường rơi vào những **nhóm chức năng: Nhập / xuất dữ liệu , thống kê, sắp xếp, liệt kê**

Ví dụ

Viết chương trình nhập số nguyên dương n và in ra màn hình các ước số của n

❖ Phân tích bài toán:

- **Input:** n (Để xác định tham số)

  Kiểu dữ liệu: số nguyên dương (*int*).

- **Output:** In ra các ước số của n (Để xác định kiểu dữ liệu trả về của hàm)

  Xuất ra màn hình → Không trả về giá trị → Kiểu dữ liệu của hàm là ***void***

- **Xác định tên hàm:** Hàm này dùng in ra các ước số của n nên có thể đặt là *LietKeUocSo*

  void LietKeUocSo(int n);

❖ Cài đặt

Hàm trả về giá trị

❖ Cài đặt

```
<Kiểu dữ liệu trả về> TênHàm([danh sách các tham số])
{
    <Kiểu dữ liệu trả về> kq;
    Khai báo các biến cục bộ
    Các câu lệnh / khối lệnh hay lời gọi đến hàm khác.
    return kq;
}
```

❖ Gọi hàm

<Kiểu dữ liệu trả về của hàm> Tên biến = TênHàm (danh sách tên các đối số);

Những phương thức này thường rơi vào các nhóm: **Tính tổng, tích, trung bình, đếm, kiểm tra, tìm kiếm**

Ví dụ

Viết chương trình nhập số nguyên dương n và tính tổng

$$S_n = 1+2+3+\cdots+n \qquad ;n>0$$

❖ Phân tích bài toán:

Input: n (Để xác định tham số)

- *Kiểu dữ liệu: số nguyên dương (int).*

Output: Tổng S (Để xác định kiểu dữ liệu phương thức)

- *Trả về giá trị của S.*
- *S là tổng các số nguyên dương nên S cũng là số nguyên dương → Kiểu trả về của hàm là int (hoặc long).*

Xác định TênHàm:

Dùng tính tổng S nên có thể đặt là ***TongS***

long ***TongS****(int n);*

❖ Cài đặt

Truyền tham số cho hàm

- Trong ví dụ trên tại sao hàm TongS khai báo đối số x TongS(int x) nhưng gọi hàm lại là n

S = TongS(n);

- Xét 2 ví dụ sau:

Xét ví dụ (1)

Hoán vị 2 số nguyên a, b cho trước

```
#include <stdio.h>
#include <conio.h>
void HoanVi(int a, int b)
{
        int tam;
        tam = a;
        a = b;
        b = tam;
        printf("Trong HoanVi: a = %d; b = %d \n",a,b);
}
void main()
{
        int a = 5, b = 21;
        printf("Truoc khi HoanVi: a = %d; b = %d \n",a,b);
        HoanVi(a, b);
         printf("Sau khi HoanVi: a = %d; b = %d \n",a,b);
}
```

a, b: tham số mặc định

hàm HoanVi không thay đổi giá trị của a và b

Xét ví dụ (2)

Hoán vị 2 số nguyên a, b cho trước

```
#include <stdio.h>
#include <conio.h>
void HoanVi(int *a, int *b)
{
        int tam;
        tam = *a;
        *a = *b;
        *b = tam;
        printf("Trong HoanVi: a = %d; b = %d \n ",*a,*b);
}
void main()
{
        int a = 5, b = 21;
        printf("Truoc khi HoanVi: a = %d; b = %d \n",a,b);
        HoanVi(&a, &b);
         printf("Sau khi HoanVi: a = %d; b = %d \n",a,b);
}
```

a, b: tham số địa chỉ của số int, khai báo với dấu *

truyền địa chỉ của a và b vào hàm HoanVi

Truyền tham số cho hàm

C hỗ trợ 2 cách truyền tham số:

- *Truyền tham số bởi giá trị (truyền giá trị - call by value) còn gọi truyền* ***tham trị***
- *Truyền tham số bởi địa chỉ (truyền địa chỉ - call by address) còn gọi truyền* ***tham biến***

Mở rộng với C++

- *Truyền tham chiếu (call by reference)*

Ví dụ: Hoán vị 2 số nguyên a, b cho trước

Truyền tham trị

- Hàm sẽ xử lý trên bản sao của tham số
- → **Hàm không thể thay đổi giá trị của tham số được.**
- Được dùng trong các trường hợp cần chuyển dữ liệu vào bên trong hàm để xử lý, tính toán
- Mặc định hàm là truyền giá trị (tham trị)
- Ví dụ hàm có sẵn của C truyền giá trị:
  - *float sqrt(float); //tính căn bậc 2*
  - *double pow(double, double); //tính lũy thừa*

Truyền địa chỉ

- Hàm sẽ xử lý trên chính tham số nhờ vào địa chỉ của chúng

→**Hàm có thể thay đổi giá trị của tham số.**

- Được dùng trong các trường hợp cần chuyển dữ liệu là kết quả xử lý được bên trong hàm ra *"ngoài"* cho các hàm khác sử dụng.
- Khai báo tham số của hàm: **Kiểu dữ liệu * tên biến**
- Ví dụ hàm có sẵn của C truyền địa chỉ (truyền tham biến)

```
int scanf(const char *format, adr1, adr2, …);
```

Truyền tham chiếu (C++)

- Khi muốn tham số hình thức và tham số thực cùng địa chỉ (bản chất là cùng ô nhớ nhưng khác tên), ta dùng cách chuyển tham chiếu cho hàm.
- Khai báo tham số của hàm: **Kiểu dữ liệu & tên biến**
- Như vậy, mọi thay đổi đối với tham số hình thức cũng làm thay đổi tham số thực.
- Có thể dùng chuyển tham chiếu để trả về giá trị cho nơi gọi hàm.

Thuật toán tráo đổi giá trị 2 biến

- **Bài toán:** Cho 2 số x và y làm thể nào để biến đổi giá trị x và y
- **Xác định bài toán:**
  - *Input: Hai biến x và y có giá trị tương ứng là a, b*
  - *Output: Hai biến x và y có giá trị tương ứng là b, a*
- **Mô tả thuật toán**

Mượn một biến z để chứa giá trị tạm thời

Bước 1: z ← x {Sau bước này giá trị của z sẽ bằng a}

Bước 2: x ← y {Sau bước này giá trị của x sẽ bằng b}

Bước 3: y ← z {Sau bước này giá trị của y sẽ bằng giá trị của z, chính là giá trị ban đầu a của biến x}

Nguyên tắc xây dựng hàm

Trước khi xây dựng hàm phải trả lời những câu hỏi sau:

- Hàm trả về gì? → Xác định kiểu dữ liệu trả về của hàm
- Hàm làm gì? → Xác định tên hàm
- Cần những thông tin gì để hàm xử lý? → Xác định tham số

Ứng với mỗi thông tin đã xác định, xác định xem đã có giá trị trước khi vào hàm chưa,

- Nếu chưa có → Tham biến (tham chiếu trong C++)
- Nếu có mà sau khi thực hiện xong hàm vẫn không thay đổi → Tham trị (không là tham biến)
- Nếu có mà sau khi thực hiện xong hàm thì giá trị cũng bị thay đổi theo → Tham biến

Ví dụ: